IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Applicant	: Wolrich, et. al.	Art Unit	: 2155
Serial No.	: 09/475,614	Examiner	: Eng, David Y
Filed	: 12/30/1999	Assignee	: Intel Corporation

Title : METHOD AND APPARATUS FOR CONTROL OF RECEIVE DATA

MAIL STOP APPEAL BRIEF - PATENTS
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

AMENDED APPEAL BRIEF

(i) Real Party in Interest

The real party in interest in this appeal is Intel Corporation, a Delaware corporation having a principal place of business at 2200 Mission College Blvd, Santa Clara, CA 95052. Intel is the assignee of the entire right, title, and interest in the above-noted application.

(ii) Related Appeals and Interferences

An appeal was filed on 8/26/2006 for U.S. serial no. 09/626,535, filed 7/27/2000, entitled "Multi-Threaded Scheduled Receive For Fast Network Port Data". The appeal brief for that application has not yet been filed.

(iii) Status of Claims

Claims 1-24 and 44 are pending and being appealed with claims 1, 17, and 18 being independent.

Claims 25-43 were previously cancelled.

(iv) Status of Amendments

No amendments were filed after the final rejection mailed on 01/19/2005.

(v) Summary of Claimed Subject Matter

Claim 1

One aspect of Appellant's invention is set out in claim 1 as a method of receiving data from a network. ***"In one aspect of the invention, receiving data from a network includes issuing a receive request directing the transfer of data from one of the plurality of device ports to a buffer memory and specifying a thread from among a plurality of processing threads to process the data."***[1]

An inventive feature of Appellant's independent claim 1 includes issuing a request directing a transfer of data from a one of a plurality of device ports to a storage unit and specifying a thread from among a plurality of processing program threads to process the data. ***"Each of the network devices attached to the I/O Bus 16 can include a plurality of ports to be serviced by the processor 12."***[2]

"Referring to FIG. 3, an exemplary thread task assignment 90 is shown. Typically, one of the microengine threads is assigned to serve as a receive scheduler 92 and another as a transmit scheduler 94. A plurality of threads are configured as receive processing threads 96 and transmit processing (or "fill") threads 98. ... The receive scheduler thread 92 assigns packets to receive processing threads 96. ... Once the receive processing thread or threads 96 has processed the packet, it either ... or stores the packet in the SDRAM and queues the packet ...."[3]

---

[1] Id., page 1, line 22, to page 2, line 4.
[2] Id., page 6, lines 14-16.
[3] Id., page 10, line 5, to page 11, line 2.

Claim 17

Another aspect of Appellant's invention is set out in independent claim 17 as a method of receiving data from a plurality of peripheral ports. ***"The receive processing threads 96 may be dedicated to servicing particular ports or may be assigned to ports dynamically by the receive scheduler thread 92."***[4]

An inventive feature of Appellant's independent claim 17 includes determining that the one of the plurality of peripheral ports requires servicing. ***"The receive scheduler 92 determines which ports need to be serviced by reading the RCV_RDY_HI, RCV_RDY_LO and RCV_RDY_CNT registers 210a, 210b and 216, respectively."***[5]

Another inventive feature of Appellant's independent claim 17 includes issuing a receive request based on the determination. ***"The RCV_RDY_CNT register 216 is one of several used by the receive scheduler to determine how to issue a receive request."***[6]

A further inventive feature of Appellant's independent claim 17 includes the receive request directing the transfer of data from the one of the plurality of peripheral ports to a buffer memory and specifies a program thread from among of a plurality of processing program threads to process the data. ***"Referring to FIG. 3, an exemplary thread task assignment 90 is shown. Typically, one of the microengine threads is assigned to serve as a receive scheduler 92 and another as a transmit scheduler 94. A plurality of threads are configured as receive processing threads 96 and transmit processing (or "fill") threads 98. ... The receive scheduler thread 92 assigns packets to receive processing threads 96. .... Once the receive processing***

[4] Id., page 11, lines 11-13.
[5] Id., page 24, lines 11-13.
[6] Id., page 20, lines 12-15,

thread or threads 96 has processed the packet, it either … or stores the packet in the SDRAM and queues the packet …"[7]

Yet another inventive feature of Appellant's independent claim 17 includes transferring the data to the buffer memory and signaling to the specified thread that the data is ready for processing. ***"The receive FIFO (RFIFO) 136 includes data buffers for holding data received from the Fbus 132 and is read by the microengines 22."***[8] ***" The RSM 166 processes the request in the RCV_REQ FIFO 230 (transaction 3). The RSM 166 responds to the request by moving the requested data into the RFIFO 136 (transaction 4), writing associated control information to the RCV_CTL FIFO 232 (transaction 5) and generating a start_receive signal event to the receive processing thread 96 specified in the receive request (transaction 6)."***[9]

Claim 18

A further aspect of Appellant's invention is set out in independent claim 18 as an article comprising a computer-readable medium which stores computer-executable instructions for receiving data from a plurality of ports. ***"The functionality of the microengine threads is determined by microcode loaded (via the core processor) for a particular user's application into each microengine's control store 70."***[10]

"In one aspect of the invention, receiving data from a network includes issuing a receive request directing the transfer of data from one of the plurality of device ports to a buffer memory and specifying a thread from among a plurality of processing threads to

---

[7] Id., page 10, line 5, to page 11, line 2.
[8] Id., page 14, lines 2-4.
[9] Id., page 25, lines 16-22.
[10] Id., page 10, lines 2-5.

process the data."[11]

An inventive feature of Appellant's independent claim 18 includes instructions causing a computer to issue a receive request directing a transfer of data from one of a plurality of device ports to a storage buffer and specifying a program thread from among a plurality of processing program threads to process the data. . ***"Each of the network devices attached to the I/O Bus 16 can include a plurality of ports to be serviced by the processor 12."***[12] ***"Referring to FIG. 3, an exemplary thread task assignment 90 is shown. Typically, one of the microengine threads is assigned to serve as a receive scheduler 92 and another as a transmit scheduler 94. A plurality of threads are configured as receive processing threads 96 and transmit processing (or "fill") threads 98. ... The receive scheduler thread 92 assigns packets to receive processing threads 96. ... Once the receive processing thread or threads 96 has processed the packet, it either ... or stores the packet in the SDRAM and queues the packet ..."***[13]

(vi) Grounds of Rejection to be Reviewed on Appeal

Claims 1-25 and 44 were rejected under 35 U.S.C. 103(a) as being unpatentable over Allison (6,373,848) in view of Belkin (U.S. 6,604,125).

---

[11] Id., page 1, line 22, to page 2, line 4.
[12] Id., page 6, lines 14-16.
[13] Id., page 10, line 5, to page 11, line 2.

(vii) Arguments

(a) Obviousness

"It is well established that the burden is on the PTO to establish a prima facie showing of obviousness, *In re Fritsch*, 972 F.2d. 1260, 23 U.S.P.Q.2d 1780 (C.C.P.A., 1972)."

In *KSR International Co. v. Teleflex Inc.*, __ U.S. __, 2007 WL 1237837 (Apr. 30, 2007), the Supreme Court reversed a decision by the Court of Appeal's for the Federal Circuit decision that reversed a summary judgment of obviousness on the ground that the district court had not adequately identified a motivation to combine two prior art references. The invention was a combination of a prior art repositionable gas pedal, with prior art electronic (rather than mechanical cable) gas pedal position sensing. The Court first rejected the "rigid" teaching suggestion motivation (TSM) requirement applied by the Federal Circuit, since the Court's obviousness decisions had all advocated a "flexible" and "functional" approach that cautioned against "granting a patent based on the combination of elements found in the prior art."

With respect to the genesis of the TSM requirement, the Court noted that although "[a]s is clear from cases such as Adams[14], a patent composed of several elements is not proved obvious merely by demonstrating that each of its elements was, independently, known in the prior art. Although common sense directs one to look with care at a patent application that claims as innovation the combination of two known devices according to their established functions, it can be important to identify a reason that would have prompted a person of ordinary skill in the relevant field to combine the elements in the way the claimed new invention does. This is so because inventions in most, if not all, instances rely upon building blocks long since uncovered, and claimed discoveries almost of necessity will be combinations of what, in some sense, is already known."

In application of the TSM requirement, the Court cautioned that: "Helpful insights, however, need not become rigid and mandatory formulas; and when it is so applied, the TSM test is incompatible with our precedents."

[14] United States v. Adams, 383 U. S. 39, 40 (1966)

"It is well established that there must be some logical reason apparent from the evidence or record to justify combination or modification of references. *In re Regal*, 526 F.2d 1399 188, U.S.P.Q.2d 136 (C.C.P.A. 1975). In addition, even if all of the elements of claims are disclosed in various prior art references, the claimed invention taken as a whole cannot be said to be obvious without some reason given in the prior art why one of ordinary skill in the art would have been prompted to combine the teachings of the references to arrive at the claimed invention. Id. Even if the cited references show the various elements suggested by the Examiner in order to support a conclusion that it would have been obvious to combine the cited references, the references must either expressly or impliedly suggest the claimed combination or the Examiner must present a convincing line of reasoning as to why one skilled in the art would have found the claimed invention obvious in light of the teachings of the references. *Ex Parte Clapp*, 227 U.S.P.Q.2d 972, 973 (Board. Pat. App. & Inf. 985)."

"The mere fact that the prior art could be so modified would not have made the modification obvious unless the prior art suggested the desirability of the modification." *In re Gordon*, 221 U.S.P.Q. 1125, 1127 (Fed. Cir. 1984).

> Although the Commissioner suggests that [the structure in the primary prior art reference] could readily be modified to form the [claimed] structure, "[t]he mere fact that the prior art could be so modified would not have made the modification obvious unless the prior art suggested the desirability of the modification." *In re Laskowski*, 10 U.S.P.Q. 2d 1397, 1398 (Fed. Cir. 1989).

"The claimed invention must be considered as a whole, and the question is whether there is something in the prior art as a whole to suggest the desirability, and thus the obviousness, of making the combination." *Lindemann Maschinenfabrik GMBH v. American Hoist & Derrick*,

221 U.S.P.Q. 481, 488 (Fed. Cir. 1984).

> Obviousness cannot be established by combining the teachings of the prior art to produce the claimed invention, absent some teaching or suggestion supporting the combination. Under Section 103, teachings of references can be combined only if there is some suggestion or incentive to do so. *ACS Hospital Systems, Inc. v. Montefiore Hospital*, 221 U.S.P.Q. 929, 933 (Fed. Cir. 1984) (emphasis in original, footnotes omitted).

"The critical inquiry is whether 'there is something in the prior art as a whole to suggest the desirability, and thus the obviousness, of making the combination.'" *Fromson v. Advance Offset Plate, Inc.*, 225 U.S.P.Q. 26, 31 (Fed. Cir. 1985).

(b) Claim Group I: Independent Claims 1 and 18.

The Examiner rejected claim 1 as being unpatentable over Allison in view of Belkin. However, neither Allison nor Belkin, alone or in combination, describe or suggest the subject matter recited by claim 1.

(1) The Examiner Has Made Unsupported Characterizations of Allison

Allison describes an adapter that channels data from multiple ports through a single Rx (receive) Media Access Control (MAC) Unit 28. The adapter services the multiple ports with the single MAC 28 by using a multiplexer 18 to select each port in turn. When selected, data from the port travels from the port, through the multiplexer 18 and Rx MII Interface 22 to the Rx MAC 28. The Rx MAC 28, in turn, writes the data into an Rx FIFO 43. Once data accumulates for the port in the Rx FIFO 43, control logic 34 moves data from the Rx FIFO 43 to a host system 12 via bus 14.

The multiplexer 18 is controlled by a port selector 46 that selects a port each chip clock

cycle (abstract). In addition to controlling which port is handled by the Rx MAC 28 by the multiplexer 18, the port selector 46 also causes different per-port state registers to be used by the Rx MAC 28 and Rx FIFO 43 for data from a selected port.

In rejecting claim 1 based on Allison, the Examiner makes a number of assertions that are unsupported by Allison. In particular, the Examiner identifies different executable instructions, subroutines, and program counters as being present in Allison. However, Allison does not make any mention of executable instructions, program counters, or subroutines. Additionally, the Examiner has built the rejection of the independent claims around an unsupported assertion regarding the functionality of what appears to be conventional flow-chart notation.

In the Office Action mailed 05/07/2004, the Examiner rejected claims 1, 17, 18 by arguing that Allison teaches:

> "Issuing a request (see Figure 9 and the description in col. 11) directing a transfer of data from one of a plurality of device ports (ports 1-n in Figure 1) to a storage unit (see registers or FIFO) and specifying (instruction program counter) a thread (instructions, see line 57, col. 2) to process (control logic 34) the data."

(page 3, office action mailed 05/07/2004).

Applicants disagree with the Examiner's assertion that Allison describes the Rx FIFO 43 as specifying an instruction program counter to the control logic 34. Neither the term nor the concept of an "instruction program counter" is described anywhere in Allison. Allison does state that the Rx FIFO 43 sends control information to the control logic 34 when data has accumulated in the Rx FIFO 43 (col. 11, lines 46-53). However, Allison does not describe the content of this control information as including an instruction program counter as asserted by the Examiner. Allison does, in one passage, refer to the control information as an "instruction" (col. 2, line 57)

which the Examiner seems to have seized on as meaning the control information represents either one of a sequence of processor executable instructions that form a program or an instruction counter. However, there is no support for the contention that the Rx FIFO 43 in Allison sends instructions of a program to the control logic for execution or that the Rx FIFO 43 specifies a program counter.

The Examiner later stated what was considered to be the request in Allison that both initiated a data transfer and would be modified to specify a thread to process the data. The Examiner stated:

> "... The request in Allison is taught in Figure 9 and demonstrated in Figure 1. In response to a request (the signal S which initiates the transfer shown in Figure 9), data is transferred from one of the ports (see ports 1-n in Figure 1) to RxFIFO for the control logic to process."

(page 3, office action mailed 1/19/05).

Thus, the Examiner identified "S" in FIG. 9 as the "issuing of a request specifying a data transfer" and presumably the request to modify in view of Belkin to include identification of a thread. Applicants disagree that the "S" in FIG. 9 is a request. In particular, FIG. 9 is a flow chart illustrating interaction between the different components of the adapter. The flow chart begins with an "S" symbol and ends with an "E" symbol, conforming to a commonplace flow chart convention. The Applicants pointed out that the "S" and "E" were traditional flow chart notation, not actual requests, and that the text of Allison did not describe the "S" or "E" in any way. The Examiner replied:

> "...although the 'S box' in Figure 9 of Allison could be served for

> indicating start of the flow chart, it also indicates a signal for directing the processor to execute the subroutine shown in FIG. 9 by pointing to the memory location which stores the first instruction of the subroutine. No inventive concept is seen"

(page 2, advisory action mailed 4/12/05).

Applicants disagree that "S" is "for directing the processor to execute the subroutine shown in FIG. 9 by pointing to the memory location which stores the first instruction of the subroutine". First, FIG. 9 does not depict a subroutine. FIG. 9 describes the different functions performed by different components in Allison. Additionally, there is no support in Allison for the assertion that "S" is a pointer to a memory location of an instruction nor does Allison include any description of any instruction pointers nor any subroutines.

In short, the Examiner's rejection amounts to a proposal to modify some entity, "S", which is not described in Allison beyond appearing at the start of a flow chart, but is for some reason assumed to be an instruction pointer. Furthermore, this instruction pointer should further be somehow modified to include identification of a thread to process data. In summary, Applicants disagree with both the Examiner's characterization of Allison and the proposed modification of "S" by the Examiner assuming, *arguendo*, that "S" is deemed a receive request.

The Examiner's position later changed to one stating that Allison actually does teach a plurality of threads albeit under a different name (see page 5 of the Examiner's Answer mailed 9/16/2005). In particular, the Examiner seemingly identifies the recited request as the collective output of Allison's port selector 46 to the different components of Allison's adaptor and the program threads as either instructions provided by the RxFIFO 43 of Allison or the different words of Allison's port state table. Appellants disagree that either the instructions provided by

the RxFIFO 43 or the words of the port state table constitute threads specified by a request. Additionally, Appellants disagree that those of skill in the art use the term "word" and "thread" interchangeably as stated by the Examiner and that Belkin would not cause one of skill in the art to begin doing so.

First, the Examiner continues to identify the control information provided by RxFIFO 43 to control logic 34 as threads. Again, as described above, Appellants do not agree that the control information provided by the RxFIFO 43 to the control logic constitutes a program thread. However, assuming purely for the sake of argument that this control information is considered a program thread, this control information is not specified by the port selector 46 output to the RxFIFO 43. The only information Allison describes as being sent by the port selector 46 to the RxFIFO 43 is the section of the RxFIFO 43 to store bytes output by the RxMAC 28 (see FIG. 8). Identifying a RxFIFO 43 memory section, however, does not constitute a request specifying a program thread.

Additionally, the claim recites "specifying a thread from among a plurality of threads". While the Examiner identifies the control information provided by the RxFIFO as a single thread, the Examiner did not identify what was being deemed the "plurality of threads". Allison does not state that multiple instructions are stored by the RxFIFO 43 for selection by the port selector 46. As such, the Examiner does not address how providing control information from the RxFIFO 43 to the control logic 34 constitutes specifying a thread from among a plurality of threads.

The Examiner next introduces a new argument that the words of the port state table constitute threads. The words of the port state table store data identifying the current state of a

port (e.g., "IDLE" or "WAIT"). Appellants agree that the port selector 46 of Allison generates a signal that selects the word from the port state table associated with a given port. However, data identifying the state of a particular port is not a program thread.

The Examiner further states that the words of Allison are "like" the threads of Belkin and states that the terms "word" and "thread" can therefore be used interchangeably. First, Appellants vigorously disagree with a general proposition that those of skill in the art use the term "word" and "thread" interchangeably or would do so even after reading Belkin. Additionally, the Examiner did not identify any portion of Belkin equating a word or entry of a table with a thread, thus the Examiner has failed to present a prima facia case of obvious with this new use of Belkin. Further, the Examiner use of Belkin relies on speculation of "if Allison requires more than one word in controlling the selected port" (see page 7 of the Examiner's Reply). Examiner speculation aside, Allison does not use more than one word per port. Finally, the words of the port entry table store data not program instructions. Thus, Appellants do not appreciate the importance that the Examiner attaches to storing more data in the state entry table or how storing more data in a table transforms the data into a thread.

While the Appellants have attempted to address the Examiner's reasoning for the sake of argument, Appellants' position remains that Allison does not describe a system with multiple program threads. Thus, Allison does not describe a request specifying a particular one from multiple program threads.

(2) One of Skill in the Art Would Not Provide the Threads of Belkin in Allison

The Examiner argues that one of skill in the art would use the multi-threaded system of

Belkin in Allison. In particular the Examiner states:

> "Allison teaches that each FIFO provides instructions (thread) to control logic for controlling transmitting and receiving data between the host system and the network. Allison does not teach plurality of threads. However, Belkin teaches a server for receiving data from a network (see Figure 1). The server has a storage for storing a pool of threads and a thread selector. In response to a request, a specific thread from a plurality of threads is selected to process the task requested by the request. Since both references are directed to transceiving data between a host and a network, it would have been obvious to a person of ordinary skill in the art to provide a pool of threads as taught by Belkin in Allison so that specific tasks such as transmitting or receiving can be respectively controlled by specific threads".

(page 3-4, office action mailed 05/07/2004).

Again, Applicants disagree with the Examiner's assertion that the Rx FIFO 43 of Allison provides a thread of program instructions to control logic 34 which the Examiner uses as a lynchpin in combining Belkin and Allison. Again, Allison provides no support for the assertion that the control logic 34 executes thread instructions issued by the Rx FIFO 43.

Applicants further disagree that one of skill in the art would be motivated to provide the threads of Belkin in the adapter of Allison. The threads of Belkin run at the host atop an Operating System (OS) on processor 604 (see FIG. 6; col. 17, line 60-col. 18, line 5) and perform "layer 5+" operations (e.g., HTTP, CGI, JAVA and so forth) in OSI protocol-stack terminology. The MAC adapter of Allison is more akin to the communication interface 618 that performs lower level (e.g., "layer 2" operations) than processor 604. The Examiner essentially proposes a far from obvious migration of functionality from the host processor 604 to the communications interface 618/ MAC adaptor of Allison that is not suggested by either Belkin or Allison.

Additionally, the motivation to move the high level, OS supported threads from a

processor 604 of Belkin to a low level MAC adapter is contrary to Allison's stated goal of reducing gate count (col. 3, line 10-14). That is, adding a programmable processor is inconsistent with reducing the circuitry count of the adaptor of Allison, particularly, when the Examiner has not provided a performance benefit for adding this considerable circuitry and complexity.

In conclusion, one of skill in the art would not be motivated to include the threads of Belkin in the adapter of Allison. Additionally, assuming for the sake of argument that such a combination was made, the Examiner has not identified which component of Allison would execute these threads or how/why the "S" would be modified (assuming, *arguendo*, that "S" actually is an instruction pointer) to include identification of a thread to process data. Thus, Applicants request withdrawal of the rejection of Claim 1 and its dependent claims, and for similar reasons, of claim 18 and its dependent claims.

(c) Claim Group II: Independent Claim 17.

Claim 17 also recites issuing a receive request. In addition, claim 17 recites the issuing the receive request is based on a determination that a port requires servicing. The claim further recites "transferring the data to the buffer memory and signaling to the specified thread that the data is ready for processing". As described above, Applicants assert that the Examiner's proposed combination of Allison and Belkin is based on unsupported characterizations of Allison and that one of skill in the art would not provide the threads of Belkin in Allison or provide the recited "receive request". Since the references do not describe or suggest the recited receive request, neither do they describe issuing the receive request based on a determination that a port

requires servicing or transferring the data to the buffer memory and signaling to the specified thread that the data is ready for processing. Applicants thus request withdrawal of the rejection of claim 17 and its dependent claims.

CONCLUSION

Given the above arguments, Applicant requests allowance of the independent claims and their corresponding dependent claims.

Please apply any required fees to deposit account 06-1050, referencing the attorney docket number shown above.

Respectfully submitted,

Date: Nov. 1, 2007

Ido Rabinovitch
Reg. No. L0080

Customer No. 20985
Fish & Richardson P.C.
Telephone: (617) 542-5070
Facsimile: (617) 542-8906

(viii) Claims Appendix

1. A method of receiving data from a network, comprising:

issuing a request directing a transfer of data from one of a plurality of device ports to a storage unit and specifying a thread from among a plurality of processing program threads to process the data.

2. The method of claim 1, further comprising:

determining if at least one of the plurality of device ports coupled to the network require service.

3. The method of claim 2, further comprising:

transferring the data to the storage unit and signaling to the specified program thread that the data is ready for processing.

4. The method of claim 2, wherein determining comprises:

interrogating the plurality of device ports to identify which of the plurality of device ports require service.

5. The method of claim 4, wherein determining further comprises:

preparing control information corresponding to those device ports identified as requiring service.

6. The method of claim 5, wherein the control information comprises indicators each associated with a device port receive FIFO in a corresponding one of the device ports.

7. The method of claim 6, wherein interrogating comprises:

polling the state of the ready flags to determine if the indicators are asserted, the assertion of the indicators indicating that the corresponding device ports have data ready for transfer.

8. The method of claim 7, wherein the indicators indicate that the associated device port receive FIFO has reached a threshold level of fullness.

9. The method of claim 8, wherein the indicators indicate that the associated device port receive FIFO stores a full network packet.

10. The method of claim 5, further comprising:

maintaining a receive ready count, the receive ready count being incremented in response to the control information being prepared.

11. The method of claim 5,wherein preparing control information further comprises:

writing a flag to a control and status register for each device port in the plurality of device ports that is determined to require service.

12. The method of claim 11, wherein issuing comprises:

obtaining the control information from the control and status register; and

selecting from each device port in the plurality of device ports having set bits in the control and status register a port for servicing.

13. The method of claim 1, further comprising:

determining which among the plurality of program threads is available; and

assigning an available program thread to process the data.

14. The method of claim 12, wherein selecting a port comprises:

using the receive ready count to determine if the ready flags reflect current status of the device port.

15. The method of claim 3, further comprising:

maintaining a receive request count for counting transfer of data to the storage unit, the receive request count being incremented by one upon the transfer of the data to the storage unit and signaling to the specified program thread.

16. The method of claim 15, wherein selecting a port further comprises:

using the receive request count to determine if the indicators reflect current status of the device ports.

17. A method of receiving data from a plurality of peripheral ports, comprising:

determining that the one of the plurality of peripheral ports requires servicing;

issuing a receive request based on the determination, the receive request directing the transfer of data from the one of the plurality of peripheral ports to a buffer memory and specifying a program thread from among of a plurality of processing program threads to process the data; and

transferring the data to the buffer memory and signaling to the specified thread that the data is ready for processing.

18. An article comprising a computer-readable medium which stores computer-executable instructions for receiving data from a plurality of ports, the instructions causing a computer to:

issue a receive request directing a transfer of data from one of a plurality of device ports to a storage buffer and specifying a program thread from among a plurality of processing program threads to process the data.

19. The article of claim 18, the article further comprises instructions causing a computer to:

determine if at least one of the plurality of device ports coupled to the network require service.

20. The article of claim 19, the article further comprises instructions causing a

computer to:

transfer the data to the storage unit and signal to the specified program thread that the data is ready for processing.

21. The article of claim 19, wherein the instructions to determine comprise instructions causing a computer to:

interrogate the plurality of device ports to identify which of the plurality of device ports require service; and

prepare control information corresponding to those device ports identified as requiring service.

22. The article of claim 21, the article further comprises instructions causing a computer to:

maintain a receive ready count, the receive ready count being incremented in response to the control information being prepared.

23. The article of claim 22, wherein the instructions to issue comprise instructions causing a computer to:

use the receive ready count to check the current status of the device port.

24. The article of 19, the article further comprises instructions causing a computer to:

maintain a receive request count for counting transfer of data to the storage unit, the receive request count being incremented by one upon the transfer of the data to the buffer memory and signaling to the specified program thread.

25. The article of claim 24, wherein the instructions to issue comprise instructions causing a computer to:

use the receive request count to check the current status of the device ports.

44. The method of claim 1, wherein the threads comprise threads provided by multiple programmable multi-threaded engines within a processor.

(ix) Evidence Appendix

No evidence is being submitted in this Appeal Brief.

(x) Related Proceedings Appendix

No decisions have been rendered by the board.